CM interface API

-1 file per table. Should be placed in ./Checkmath/misc folder.
*-File should be named **TBL$table_num** (e.g. TBL1, TBL2, TBL9999…)*
-Output is saved in ./Checkmath/output folder. The name is output$table_num
-Input doesn't have to be sequential. You can start from the start of the hand, or choose to make a request when needed (e.g after 3bet Preflop… or just Postflop… or after cbet on the flop etc)
-Input should only be made on hero's turn!
-In order to save output, select the appropriate checkbox in API settings. While saving output you can still view sims in Checkmath hand viewer or choose not to.
-Make sure to check out auto-recalculation rules in API settings, these are extremely helpful to get the max out of Checkmath API.
-After the tree is recalculated, it will be used for all further requests of this hand (handNum).

INPUT (JSON FORMAT)

-**myCards** - String, hero's cards e.g. As2c, 8d9h

-flopCards - String, flop (no turns or rivers anytime in the hand!). If Preflop, leave it blank or don't include this field at all. Example: 9sAhKd

-**stackSize** - String, stack size for the start of the hand (before blinds are posted). Should end with "bb". Should contain "-HU" in the end for HU-cash. Examples: 105bb, 12.5bb, 205bb-HU.

-**defender** - String, state that represents complete Preflop action leading to the Postflop. If Preflop, leave it blank or don't include this field. Example: f:f:b22:b76:f:f:c (for 6max cash), b25:c (for HU cash), c:b400:c (for Hu-SNG), f:b250:c: (for Spins-3way).
'f' represents fold, 'c' for check or call, 'bX' for bet X amount. **Important!** Bets must be brought to BB=10 for cash games and BB=100 for spins/husng!

-**gameType** - String, enum {cash, spins, sng}. For HUSNG and HU part of the Spins game use sng. For 3way Spins - spins

-**state** - String, state of the requested node, must start with “r:0”. After transition to the Postflop, the state should be reset! Examples: r:0:f:f:b25 (6max), r:0:b25:b100 (HU)

-**position** - String, hero’s position at the table. Enum {EP, MP, CO, BTN, SB, BB}. For HU use SB/BB structure

-**realCurrentPot** - *Optional.* Integer, used for recalculation, must be brought to BB=10 for cash and BB=100 for spins/husng. Use this only if you can't use realStreetStartPot.

-**realCurrentEffective** - *Optional.* Integer, used for recalculation, must be brought to BB=10 for cash and BB=100 for spins/husng. Use this only if you can't use realStreetStartEffective.

-**realStreetStartPot** - *Optional.* Integer, used for recalculation, preferred way of sending pot/effective.

-**realStreetStartEffective** - *Optional.* Integer, used for recalculation, preferred way of sending pot/effective.

-**bbSize** - *Optional.* Float. Only for cash, size of the BB for strategy output. (bets in the state must still be in BB=10 format!)

-**handNum** - String, current hand name/num for this table. New hand must have a new handNum, otherwise **may lead to unpredictable results**.

-recalc - *Optional.* String, template name. Gives the instruction to recalculate the tree from the start of the current street with a given template name. If template name is not found for this spot (street and spr), default template will be used. If no templates are found, recalculation will not happen. You can create/customize templates for each street/spr in Settings -> Recalculation ver2.

EXAMPLES **PREFLOP**


```
{"myCards": "AdKd", "stackSize": "105bb", "gameType": "cash", "state": "r:0:b22:f:c",
"position": "BTN", "bbSize" : 2, "handNum": "1"}

{"myCards": "AdKd", "stackSize": "145bb", "gameType": "cash", "state": "r:0",
"position": "EP", "bbSize" : 2, "handNum": "2"}

{"myCards": "AdJd", "stackSize": "44bb", "gameType": "cash", "state":
"r:0:f:f:f:b30:f", "position": "BB", "bbSize" : 2, "handNum": "3"}

{"myCards": "AdKd", "stackSize": "202bb-HU", "gameType": "cash", "state":
"r:0:b25:b125", "position": "SB", "bbSize" : 10, "handNum": "4"}

{"myCards": "AdKc", "stackSize": "13.5bb", "gameType": "sng", "state": "r:0:c:b300",
"position": "SB", "handNum": "5"}

{"myCards": "AdKc", "stackSize": "13.5bb", "gameType": "spins", "state": "r:0:b200:f",
"position": "BB", "handNum": "6"}
```


EXAMPLES **POSTFLOP**


```
{"myCards": "AdKd", "stackSize": "102bb", "gameType": "cash", "state": "r:0",
"position": "BB", "bbSize" : 2, "handNum": "1", "defender": "b22:f:f:c:f:b150:c:f",
"flopCards": "Ks9d2c"}

{"myCards": "AdKd", "stackSize": "102bb", "gameType": "cash", "state":
"r:0:b98:c:Qd:c", "position": "BB", "bbSize" : 2, "handNum": "1", "defender":
"b22:f:f:c:f:b150:c:f", "flopCards": "Ks9d2c"}

{"myCards": "AdKd", "stackSize": "102bb", "gameType": "cash", "state":
"r:0:b98:c:Qd:b100:c:Ah:c:b817", "position": "BB", "bbSize" : 2, "handNum": "1",
"defender": "b22:f:f:c:f:b150:c:f", "flopCards": "Ks9d2c", "realStreetStartEffective"
: 817, "realStreetStartPot": 800, "recalc": "default"}

{"myCards": "AdKd", "stackSize": "202bb-HU", "gameType": "cash", "state": "r:0:c",
"position": "SB", "handNum": "2", "defender": "b25:c", "flopCards": "Ks9d2c"}
```

{"myCards": "AdKd", "stackSize": "11.4bb", "gameType": "sng", "state": "r:0:b100", "position": "SB", "bbSize" : 2, "handNum": "3", "defender": "c:c", "flopCards": "Ks9d2c"}

{"myCards": "AdKd", "stackSize": "23bb", "gameType": "spins", "state": "r:0:c", "position": "BTN", "handNum": "4", "defender": "b200:f:c", "flopCards": "Ks9d2c"}

OUTPUT (JSON FORMAT) - *optional*

-requestState - requested state (to make sure you are looking at the right response)

-requestDefender - requested defender(to make sure you are looking at the right response)

-requestHandNum - requested hand num(to make sure you are looking at the right response)

-requestMyCards - requested hero cards (to make sure you are looking at the right response)

-requestRecalc - requested recalc field (to make sure you are looking at the right response)

-target_state - closest state your input was mapped to

-effectiveStack - String, current effective stack from the closest sim. Will be a real effective stack if the tree is recalculated.

-pot - String, current pot from the closest sim. Will be a real pot if the tree is recalculated.

ip_eq_average - String, range equity for IP player on a given node (only for postflop)

oop_eq_average - String, range equity for OOP player on a given node (only for postflop)

strategy - array of nodes (for the preflop only RNG choice is given!)

node:

-name - String, action description (RAISE 125 (50% psb), CHECK etc)
-comboFrequency - String, your hand strategy % for this node
-rangeFrequency - String, your range strategy % for this node
-rngChoice - Integer. 0/1, built-in RNG.

simType - String, enum {CLOUD, RECALC, LOCAL}. What sim was used to generate output. Local is currently unavailable…

error - String, will be present only when error occurred with its description.

Other possible lines in the output file:

"Recalculation have started: requested"
"Recalculation have started: auto-triggered"
"Recalculation have finished"